

Index Optimization with Combination of Text Clustering by Table based KNN and AHC Algorithm

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the table based KNN algorithm and the table based AHC algorithm for automating the index optimization by combining it with the text clustering. In the previous work, even if the KNN version was proposed as an approach to the index optimization, it was required to present a domain as an input text tag manually for carrying out the task. In this research, text subgroups are built based on their contents by clustering texts, and each word in a novice text is classified into one of the three categories by a classifier which corresponds to its most similar cluster. The table based AHC algorithm, and the table based KNN algorithm are adopted respectively as the approaches to the text clustering and the index optimization. The goals of this research is to automate the keyword extraction and to improve the performance of the both tasks, using the table based algorithms.

I. INTRODUCTION

The index optimization is defined as the process of optimizing the index through both the index expansion and the index filtering. The important classes are defined to words, to decide whether to use the index expansion or the index filtering for each word. The index optimization is interpreted into a classification task where each word is classified into one of expansion, inclusion, and removal. The index expansion is applied to words which are classified into expansion, and the index filtering is applied to ones which are classified into removal. The research scope is restricted to the classification task, excluding the treatment to classified words.

This research is motivated by the necessity of deciding domains as a list in designing the index optimization system. The index optimization is interpreted into a classification task domain by domain. The process of designing the index optimization system is to predefine domains, collect sample examples domain by domain, and allocate a classifier for each domain. One more motivation of this research is the successful results from applying the table based KNN algorithm and the table based AHC algorithm, respectively, to the index optimization and the text clustering. This research is intended to automate the domain decision by combining the index optimization and the text clustering with each other.

The idea of this research is to automate the domain predefinition by the text clustering, implement index op-

timization system domain by domain, and apply the table based KNN algorithm and the table based AHC algorithm to both tasks. In this research, it is assumed that the text collection is initially prepared, and each text in the collection is associated with words which are classified into one of the three classes. The similarity metric between tables is defined, and the KNN algorithm and the AHC algorithm are modified into the table-based versions, based on the similarity metric. The collection is segmented into clusters, each of which contains content based similar texts, and an index optimization system is implemented for each cluster. A domain is automatically assigned to an input text by its similarities with the cluster prototypes for selecting a corresponding classifier.

Let us mention some benefits from this research. The main problems in encoding words and texts into numerical vectors are prevented by encoding them into tables. The performances of the two tasks, such as the index optimization and the text clustering, are improved by adopting the table-based versions of the KNN algorithm and the AHC algorithm. We do not need prior knowledge and subjective views by predefining automatically domains through the text clustering. We do not need to present a domain for using the text summarization system.

Let us mention some remaining tasks as the further discussions on this research. We modify more advanced machine learning algorithms and deep learning algorithms into the table-based versions. It is necessary to define and characterize mathematically more operations on tables. We implement the index optimization system which relates to the text clustering as a real program or a library. The index optimization system may be installed in the text classification system or the text clustering system for improving their performances.

II. PROPOSED APPROACH

A. Similarity Metric

This section is concerned with the table as the representation of a word. The table is defined as a set of entries, each of which consists of an element and its weight. The table which represents a word consists of entries each of

which contains a text identifier and its weight. The similarity between tables is computed based on shared words as the semantic similarity between words. This section is intended to describe the process of encoding a word into a table, and the process of computing the similarity between tables.

The process of encoding words into tables is illustrated in Figure 1. In the inverted indexing, each word is linked with texts which include itself. In the text-word weighting, its weights in the lined texts are computed; a weight indicates a relationship between a word and a text. Entries with their lower weights are removed for downsizing the table. Refer to [1] for studying the encoding process in more detail.

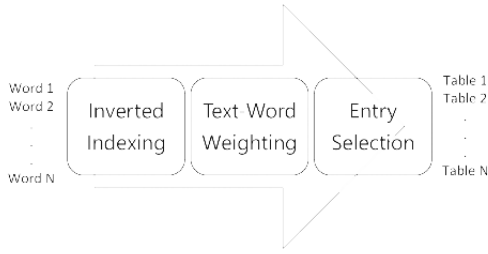


Figure 1. Process of Encoding Words into Tables

Let us mention the function of a table which maps it into a set of text identifiers as shown in Figure 2. The table which represents a word is expressed as entry set as shown in equation (1),

$$T = \{(d_1, w_1), (d_2, w_2), \dots, (d_{|T|}, w_{|T|})\} \quad (1)$$

where d_i is a text identifier and w_i is the weight of the word, T , in the text, d_i . The function for generating a list of text identifiers is expressed as equation (2),

$$F(T) = \{d_1, d_2, \dots, d_{|T|}\} \quad (2)$$

The function is the operation which takes only text identifiers without their weights as a set. The operation is applied to computing the similarity between tables which represent words.

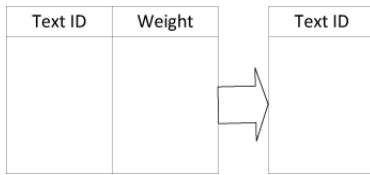


Figure 2. Conversion of Table into Text Set

Let us mention the process of computing the similarity between two tables as a semantic similarity between words. The two tables which represent words are notated respectively by $T_1 = \{(d_{11}, w_{11}), (d_{12}, w_{12}), \dots, (d_{1|T|}, w_{1|T|})\}$ and $T_2 = \{(d_{21}, w_{21}), (d_{22}, w_{22}), \dots, (d_{2|T|}, w_{2|T|})\}$. The

intersection between the two sets which is generated by applying the function, $F(\cdot)$ as shown in equation (3),

$$F(T_1) \cap F(T_2) = \{d_{s1}, d_{s2}, \dots, d_{s|F(T_1) \cap F(T_2)|}\} \quad (3)$$

and the table with entries each of which consists of a shared text identifier, d_{si} , its weight from the table, T_1 , w_{si}^1 , and its weight from the table, T_2 , w_{si}^2 , is expressed as equation (4),

$$ST_{12} = \{(d_{s1}, w_{s1}^1, w_{s1}^2), (d_{s2}, w_{s2}^1, w_{s2}^2), \dots, (d_{s|F(T_1) \cap F(T_2)|}, w_{s|F(T_1) \cap F(T_2)|}^1, w_{s|F(T_1) \cap F(T_2)|}^2)\} \quad (4)$$

The similarity between tables, T_1 and T_2 is computed by equation (5),

$$sim(T_1, T_2) = \frac{2(\sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{si}^1 + \sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{si}^2)}{\sum_{i=1}^{|T_1|} w_{1i} + \sum_{i=1}^{|T_2|} w_{2i}}. \quad (5)$$

The value which is computed from equation (5) is always given as a normalized value between zero and one as shown in equation (6),

$$0 \leq sim(T_1, T_2) \leq 1. \quad (6)$$

Let us make some remarks on the encoding process and the computation of similarity between words. A word is encoded into table by the inverted indexing, the text identifier weighting, and the entry selection. A table is expressed as a set of entries and filtered into a set of text identifiers by the function. The similarity between tables is computed based on their shared entries by equation (5). In the future research, we consider representing a word into multiple tables.

B. Table based KNN Algorithm

This section is concerned with table based KNN algorithm. It was initially proposed as the approach to the word classification by Jo in 2018 [1]. The similarity metric between tables which was described in the previous section is used for computing the similarity between a novice table and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the table-based version of KNN algorithm.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into tables, and they are notated by a set $Tr = \{T_1, T_2, \dots, T_N\}$. A novice word is encoded into a table notated by T . Its similarities with the tables which represent the sample words are computed by equation (5), as $sim(T, T_1), sim(T, T_2), \dots, sim(T, T_N)$. Some training

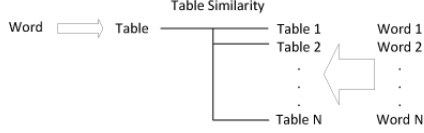


Figure 3. Similarities of Novice Input with Training Examples

examples which are most similar as the novice input are selected as its nearest neighbors.

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (7),

$$Nr = \text{Select}_k(Tr, T), Nr \subseteq Tr \quad (7)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (8),

$$Nr = \{T_1^{near}, T_2^{near}, \dots, T_k^{near}\} \quad (8)$$

The elements in the set, Nr , are used for voting their labels in the next step.

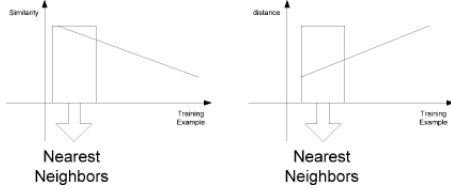


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(T_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, T , is decided by equation (9),

$$\text{label}(T) = \underset{j=1}{\text{argmax}}^m \text{Count}(Nr, c_j) \quad (9)$$

The process of deciding the label of a novice item by equation (9), is called voting.

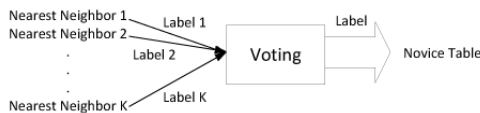


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the table-based version of KNN algorithm. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the index optimization system which adopts the table based KNN algorithm. In Section II-B, we described the proposed version of KNN algorithm as the approach to the index optimization. It is viewed as a classification task which classifies a word into one of the three categories. This section is intended to describe the index optimization system with respect to its function and its architecture.

The process of collecting the same words and encoding them into tables for implementing the index optimization system is illustrated in Figure 6. The index optimization is interpreted into a domain dependent classification; even a same word is classified differently, depending on the domain. For each domain, an independent classification task is defined. The several domains are decided, and the words which are labeled with one of the three categories exclusively in each domain. It is required to present the domain from a text which is given as the input for doing this task.

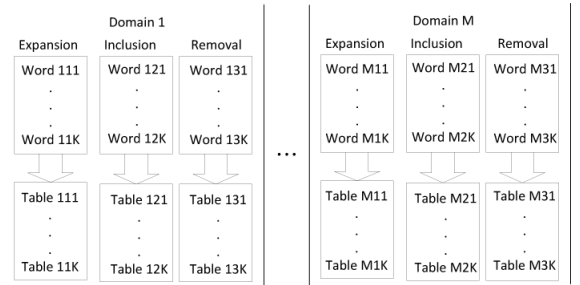


Figure 6. Preparation of Training Examples

The entire architecture of the proposed index optimization system is illustrated in Figure 7. A text is given as the input, and words are extracted from it in the indexing module. In the encoding module, the sample words in the groups, expansion, inclusion, and removal and ones which are indexed from the text are encoded into tables. In the similarity computation module and the voting module, the words which are indexed from the text are classified into one of the three categories. The words which are classified into removal are discarded in this system.

The execution process of the proposed system is illustrated as a block diagram in Figure 8. Sample words which are labeled with one of the three categories are collected

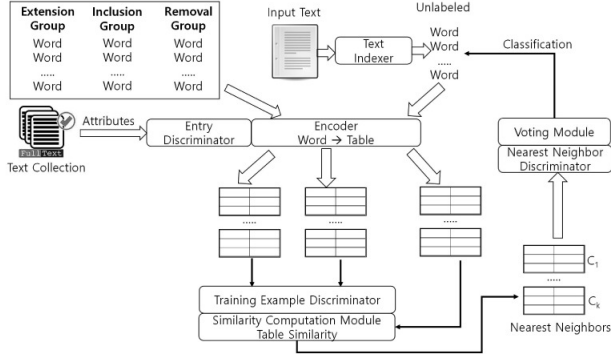


Figure 7. System Architecture

within a domain, and they are encoded into tables. An input text is indexed into a list of words, and they are also encoded into tables. The nearest neighbors are selected by the similarity computation, and the label is decided by voting ones of their nearest neighbors for each word. Ones which are classified into expansion require their associated words from external sources, ones which are classified into inclusion are preserved, and ones which are classified into removal are excluded.

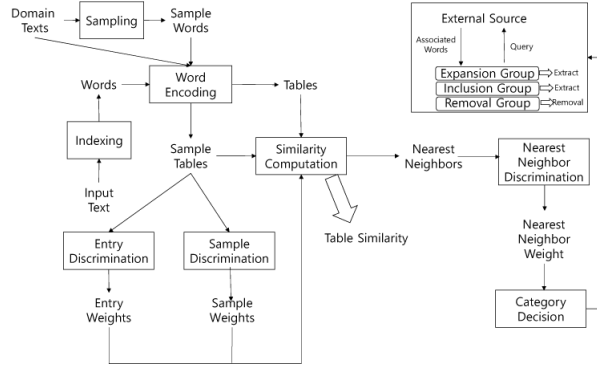


Figure 8. System Flow

Let us make some remarks on the proposed system which is illustrated in Figure 7 as its architecture. The index optimization is defined as a multiple classification task where each word is classified into expansion, inclusion, or removal. The words are encoded into tables instead of numerical vectors, and they are classified directly. The words which are classified into removal are excluded from index of the input text. In implementing the system as its complete version, we need to add the module which retrieve more words which are relevant to ones which are labeled with expansion.

D. Connection with Text Clustering

This section is concerned with the connection of the index optimization with the text clustering. In the previous

section, we mentioned the index optimization as an instance of domain dependent classification. The domains are automatically constructed from the text collection by clustering texts. The AHC algorithm which is proposed in [2] is used for implementing the text clustering. This section is intended to describe the index optimization system which is connected with the text clustering for automating the construction of domains.

The architecture of the text clustering system which was proposed in [2] is illustrated in Figure 9. The texts in the group are encoded into tables, and the AHC algorithm which is proposed in [2] is adopted as the approach. It starts with singletons as many as items, computes the similarities of all possible cluster pairs, and merge the cluster pair with its highest similarity into one cluster. The two steps are iterated until the number of clusters reach the desired number. We may consider replacing the AHC algorithm by its variants for improving the speed and the performance.

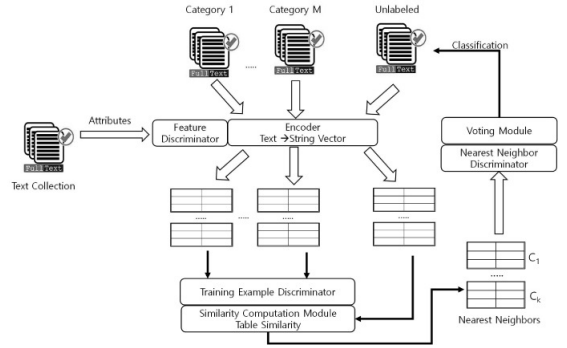


Figure 9. Text Clustering System

The connection of the index optimization with the text clustering is illustrated in Figure 10. The M domains are defined by clustering texts in a group, initially and automatically. A novice text is given as the input, and its domain, domain D , is decided by its similarities with domains. A classifier which corresponds to domain D is nominated and classify each word in a novice text into expansion, inclusion, or removal. The M index optimization systems are viewed as independent ones, each of which classifies each word into one of the three categories.

The execution flow of index optimization which is connected with the text clustering is illustrated in Figure 11. Unlabeled texts are clustered into subgroups, each of which is a domain. Sample words which are labeled with expansion, inclusion, or removal are generated from each domain. These sample words are provided for training the classifier in each index optimization system. Each classifier is used for judging the importance level of each word which is indexed from the input text.

Let us make some remarks on the connection of the index optimization with the text clustering. It is the process of segmenting a text group into subgroups based on their

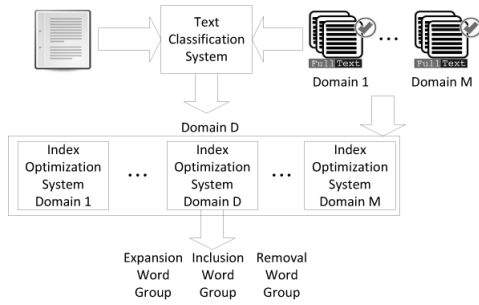


Figure 10. Index Optimization System connected with Text Clustering

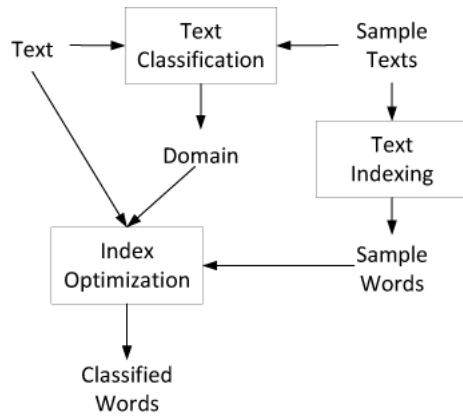


Figure 11. System Flow of Index Optimization connected with Text Clustering

content-based similarities. The role of text clustering is to define the domains initially in the architecture of connecting the index optimization with the text clustering. In each domain, sample words are generated, and its corresponding classifier is trained with them. In the future research, we consider using the index optimization for clustering texts in the opposite direction.

REFERENCES

- [1] T. Jo, "Table based K Nearest Neighbor for Word Categorization in News Articles", The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.
- [2] T. Jo, "Applying Table based AHC Algorithm to News Article Clustering", 8-11, The Proceedings of International Conference on Green and Human Information Technology, Part I, 2019.
- [3] T. Jo, "Optimizing Index of News Articles by Table based Version of K Nearest Neighbors", The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.